Seasonal and Non Seasonal Garch Time Series analysis

MA-641A TIME SERIES ANALYSIS

By Ujwala Kamineni

Seasonal: Time series analysis for Advance Monthly Sales for Retail and Food Services

Abstract :

This report provides a comprehensive analysis of the "Advance Monthly Sales for Retail and Food Services" dataset, covering the period from 1992 to 2024. The analysis aims to uncover the overall sales trends, seasonality, and the impact of significant economic events on retail sales. By cleaning and preparing the data, key insights were obtained, revealing steady growth in the retail sector despite the disruptions caused by the 2008 financial crisis and the COVID-19 pandemic. Seasonal patterns are evident, with peaks around the holiday seasons and a consistent post-holiday lull. This report identifies long-term trends, provides insights into the resilience of the retail sector, and highlights areas of significant impact, offering valuable guidance for future market strategies and policy-making.

Introduction:

Retail and food services form a significant portion of the U.S. economy, providing insights into consumer behavior and economic health. The "Advance Monthly Sales for Retail and Food Services" dataset, published by the U.S. Census Bureau, offers monthly estimates of retail sales across various sectors. By analyzing the data from 1992 to 2024, this report aims to elucidate the overall trends, seasonality, and the effects of economic downturns on the retail industry.

The dataset includes the monthly sales figures in millions of USD, which allows for an analysis of the cumulative effects of various economic events. Notably, the 2008 financial crisis and the COVID-19 pandemic have led to significant disruptions in consumer spending, resulting in sharp declines in retail sales. This analysis intends to compare pre- and post-crisis periods, uncover seasonality patterns, and identify long-term trends.

Through data preparation, visualization, and comparison, this report provides valuable insights into the behavior of retail sales, the resilience of the retail industry, and the influence of economic events on consumer spending. Ultimately, it aims to contribute to strategic planning, policy-making, and market understanding for stakeholders within the retail sector.

Data Description:

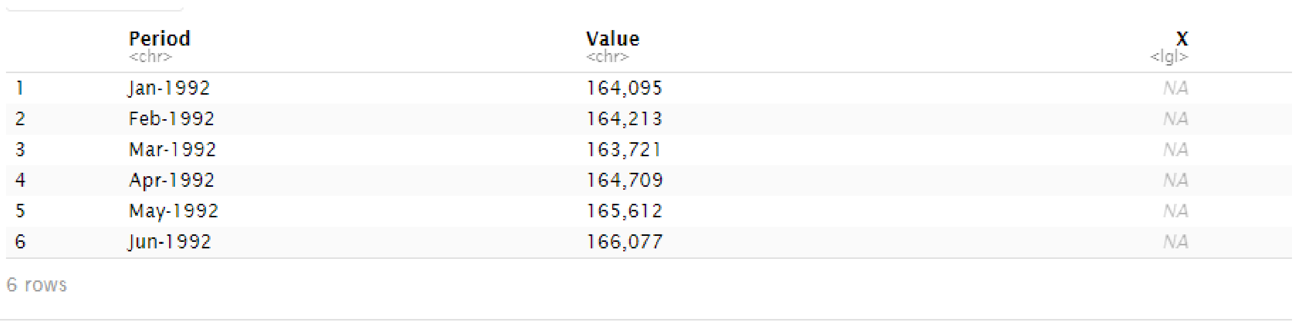
Date Range: From January 1992 to December 2024

Datasource Description: The data is from the U.S Census Bureau website and can be accessed using the link: https://www.census.gov/econ/currentdata/?programCode=MARTS&startYear=1992&endYear=2024&categories[]=44X72&dataType=SM&geoLevel=US&adjusted=1&notAdjusted=0&errorData=0

Dataset Description: The dataset is sourced from the Advance Monthly Sales report for Retail and Food Services. It includes information on retail trade and food services, covering data from 1992 to 2024. The data was extracted on April 29, 2024. The dataset has 403 rows and 3 columns.

Dataset Entries: Three columns were Period, Value and X

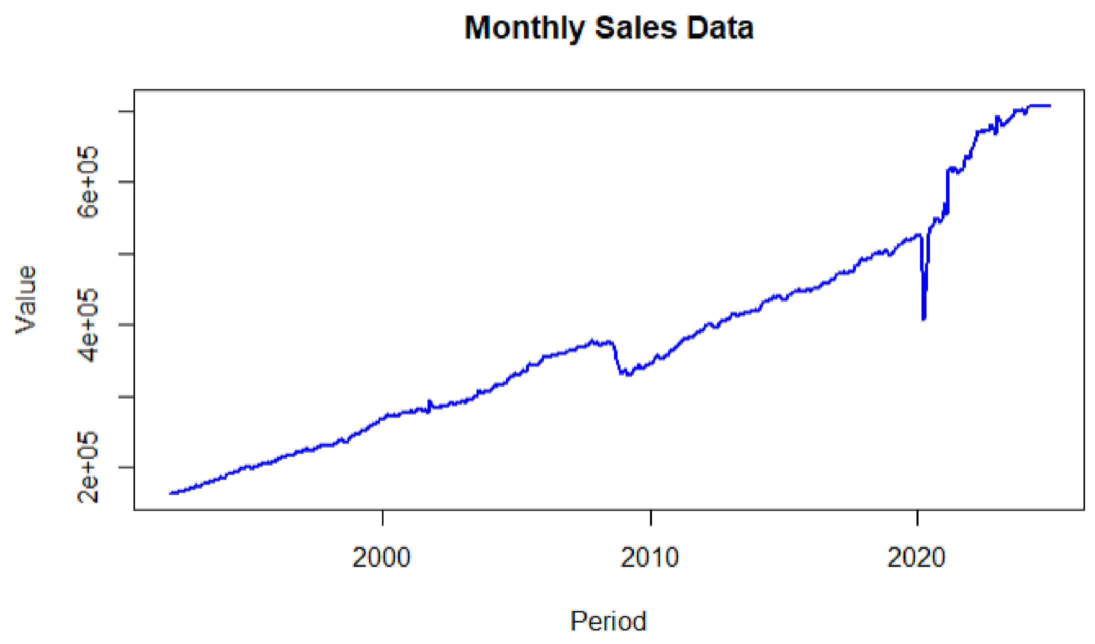
Initial Look of the Dataset: This dataset provides a comprehensive view of the retail and food services industry over several decades. However, further inspection and data cleaning will be required to understand the exact nature and structure of the content, including any patterns or trends.



	Period <chr>	Value <chr>	X <lgl>
1	Jan-1992	164,095	NA
2	Feb-1992	164,213	NA
3	Mar-1992	163,721	NA
4	Apr-1992	164,709	NA
5	May-1992	165,612	NA
6	Jun-1992	166,077	NA

6 rows

The plot of monthly sales of Retail and Food services is as follows:



- **X-Axis (Period):**

  The x-axis represents a period from approximately 1992 to 2024, indicating that the data covers over three decades.

- **Y-Axis (Value):**

  The y-axis represents the sales value, reaching up to about 600,000.

- **Trend:**

The general trend is upward, suggesting consistent growth in monthly sales over time. There is a noticeable dip around 2020, likely reflecting the global economic impact during that period (such as the COVID-19 pandemic). After his dip, the trend continues upward, showing recovery and further growth.

Data Pre-processing:

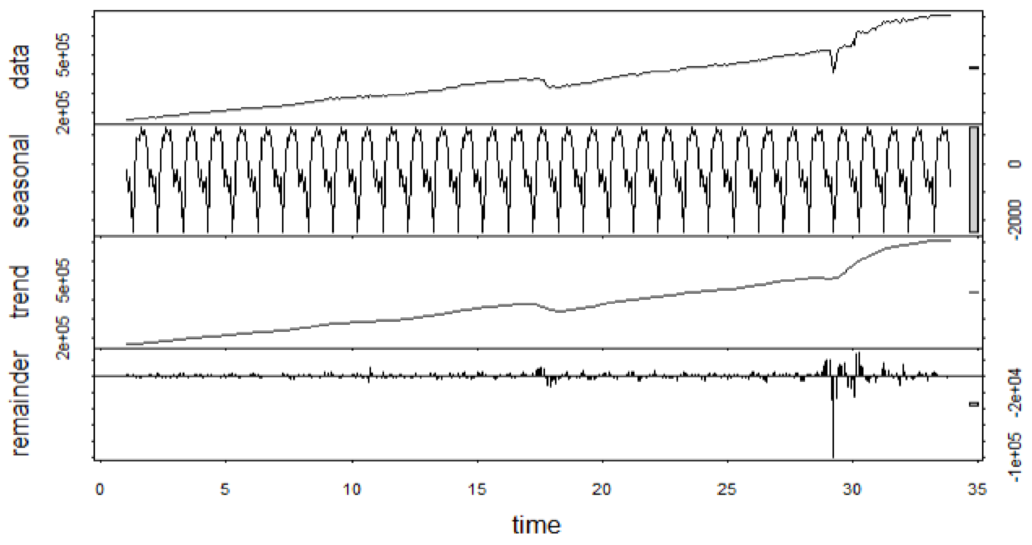
Before any further preprocessing, we want to remove the null values. In the process of data cleaning, I have used interpolation to handle missed values.

```
# Reading the data
data <- read.csv("C:/Users/Lenovo/Downloads/SeriesReport-202404290224-V.csv", stringsAsFactors = FALSE, skip=7)
head(data)

data$Period <- as.Date(data$Period, format="%b-%Y")
data$Value <- as.numeric(gsub(",", "", data$Value))

# Handling missing values by interpolation
data$Value <- na.interp(data$Value)
```

DECOMPOSING THE TIME SERIES:

Decomposing the time series to have a look at the seasonal components, trend components, and residuals in it.

```
# Seasonal Decomposition
data_ts <- ts(data$Value, frequency=12)
decomp <- stl(data_ts, s.window="periodic")
plot(decomp)
```



Original Data (Top Panel): This plot shows the overall trend of the time series, including all components.

Seasonal (Second Panel): The seasonal plot identifies repeating patterns that recur at regular intervals. In this case, the seasonal component has a regular cycle that is repeated yearly.

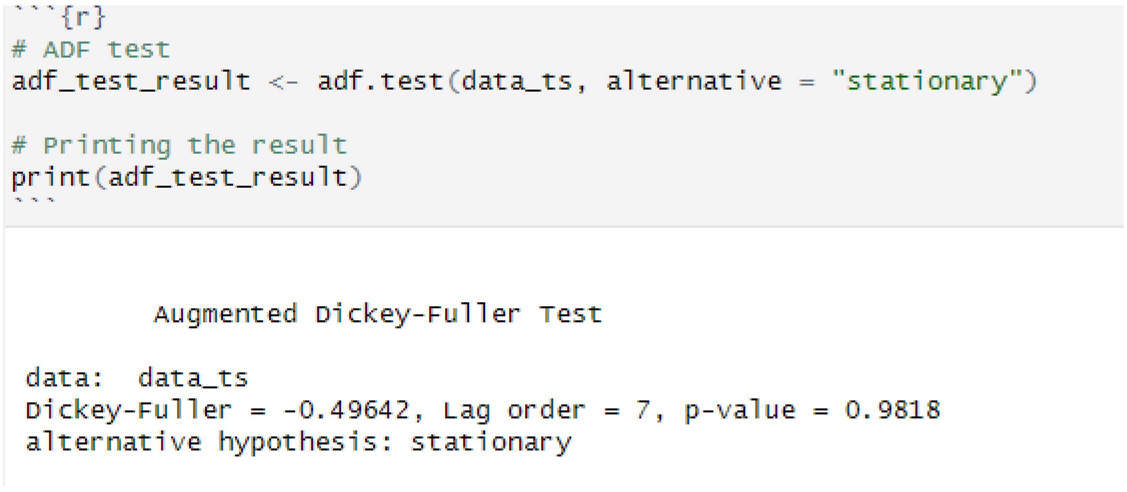
Trend (Third Panel): The trend plot smooths out short-term fluctuations, revealing the long-term movement of the series. There's a noticeable dip around the 30th unit, which is likely the impact of a specific event.

Remainder (Residuals, Bottom Panel): This plot shows the remaining noise after removing the trend and seasonal components. It captures the random variations that aren't explained by seasonality or trend.

FITTING THE TIME SERIES MODEL

Stationarity Check :

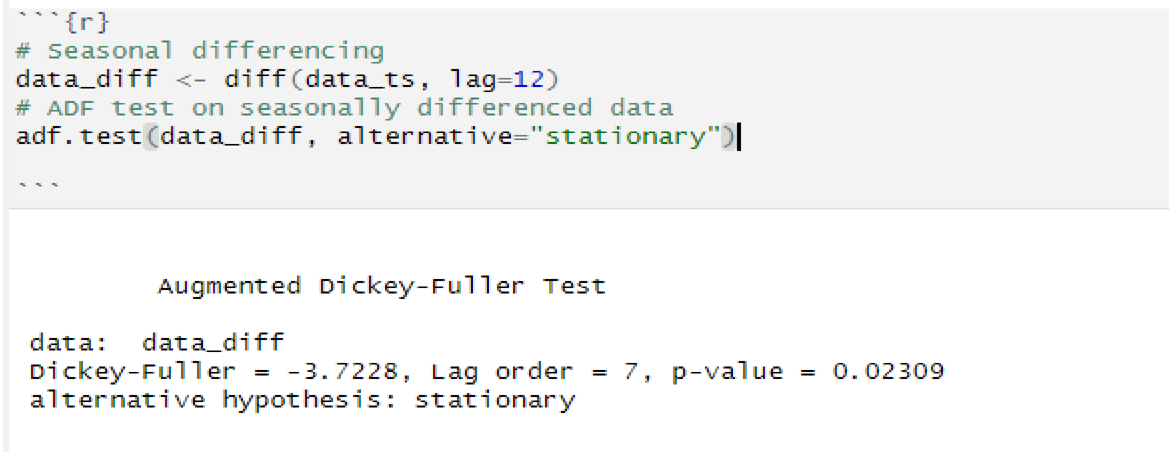
Let's start by checking if the time series is stationary or not. To do so we are going to use the Dickey-Fuller and/or augmented Dickey-Fuller test.



~~~
```{r}
# ADF test
adf_test_result <- adf.test(data_ts, alternative = "stationary")

# Printing the result
print(adf_test_result)
```
~~~

```

	Augmented Dickey-Fuller Test

data:  data_ts
Dickey-Fuller = -0.49642, Lag order = 7, p-value = 0.9818
alternative hypothesis: stationary
```

Since the p-value is greater than 0.05 indicating non-stationarity, differencing the data and checking again for stationarity.



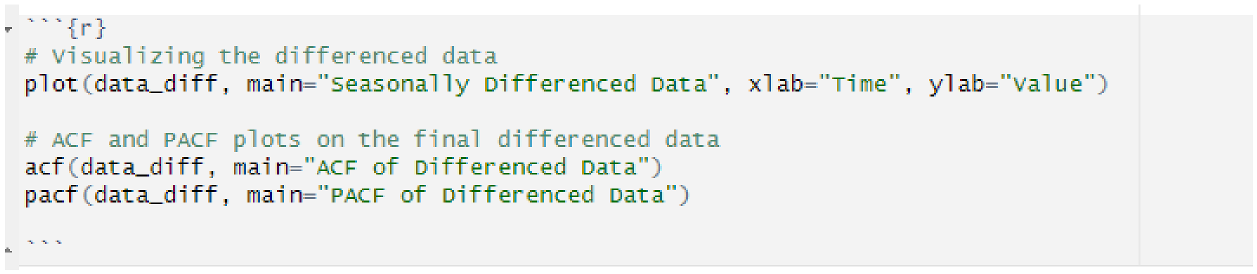
~~~
```{r}
# Seasonal differencing
data_diff <- diff(data_ts, lag=12)
# ADF test on seasonally differenced data
adf.test(data_diff, alternative="stationary")

```
~~~

```

	Augmented Dickey-Fuller Test

data:  data_diff
Dickey-Fuller = -3.7228, Lag order = 7, p-value = 0.02309
alternative hypothesis: stationary
```

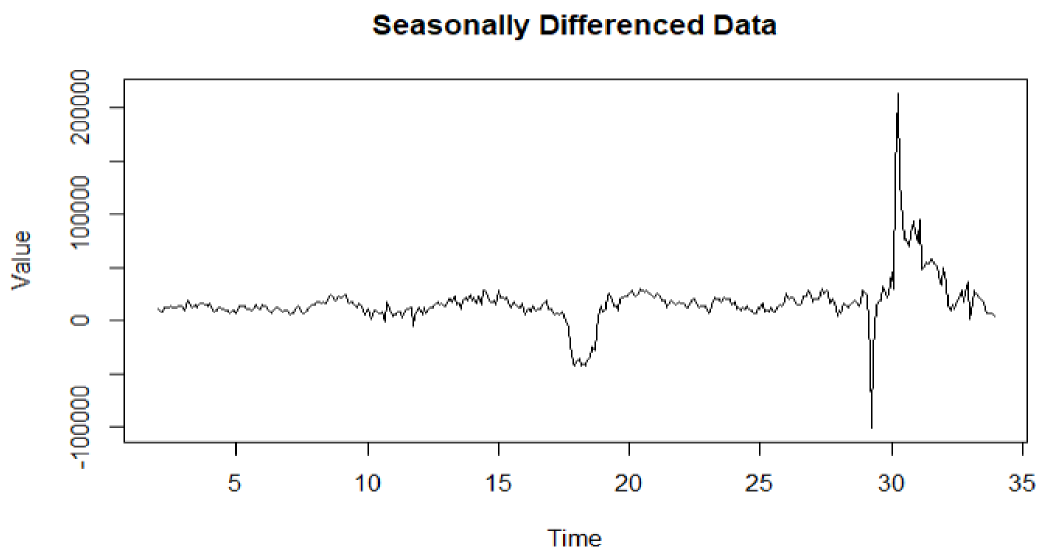
Now the p-value is less than 0.05 indicating it stationary

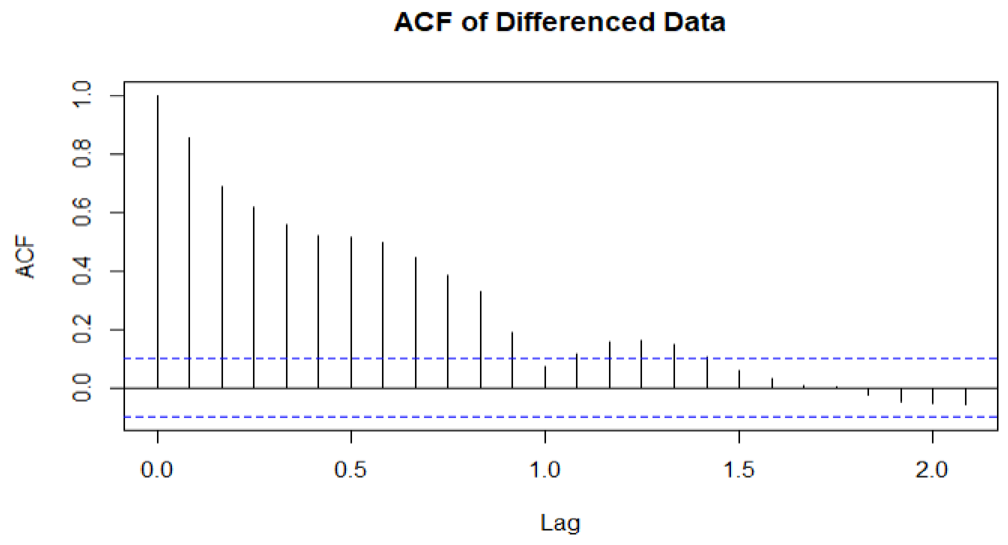
It is important to make the series stationary before fitting a model to it, because, we observe a single run of a stochastic process rather than repeated runs of the stochastic process. Hence, we need stationarity and ergodicity so that observing a long run of a stochastic process is similar to observing many independent runs of a stochastic process.

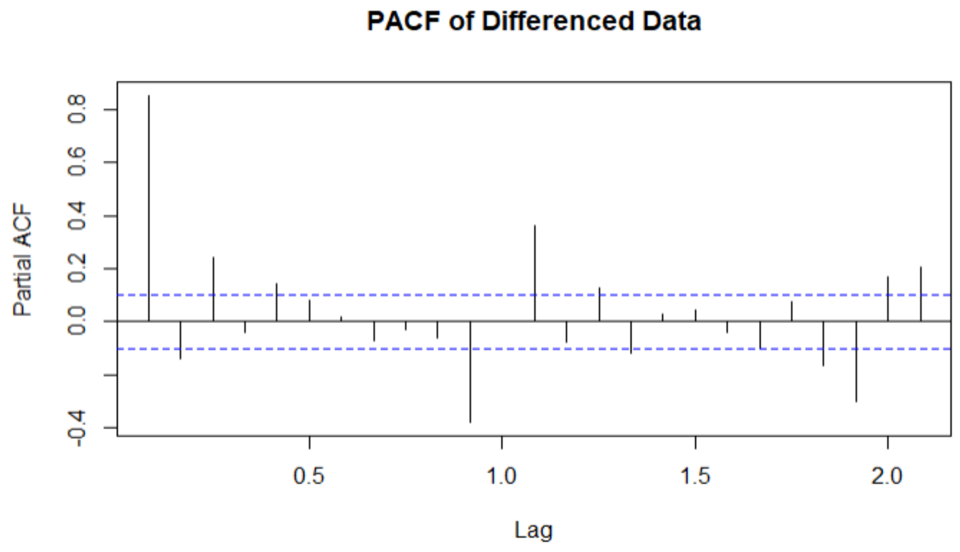
Plotting the Differenced data, ACF and PCF of the series.



~~~
```{r}
# Visualizing the differenced data
plot(data_diff, main="Seasonally Differenced Data", xlab="Time", ylab="Value")

# ACF and PACF plots on the final differenced data
acf(data_diff, main="ACF of Differenced Data")
pacf(data_diff, main="PACF of Differenced Data")

```
~~~



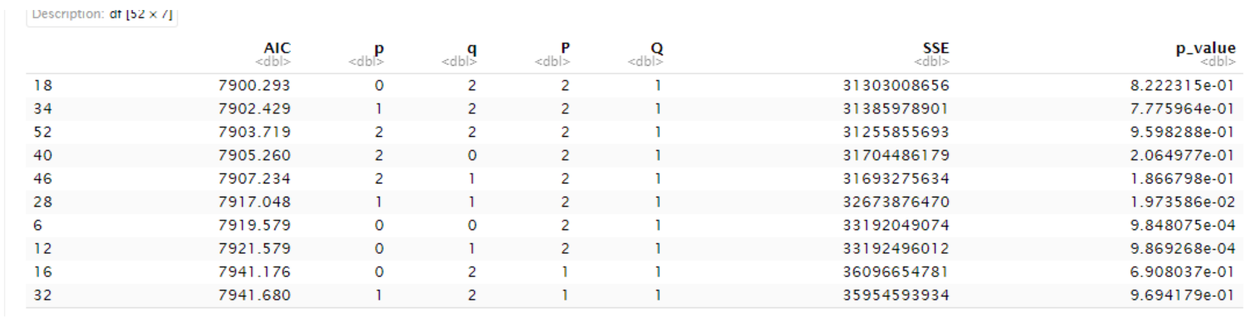




DETERMINING THE ORDER OF THE MODEL

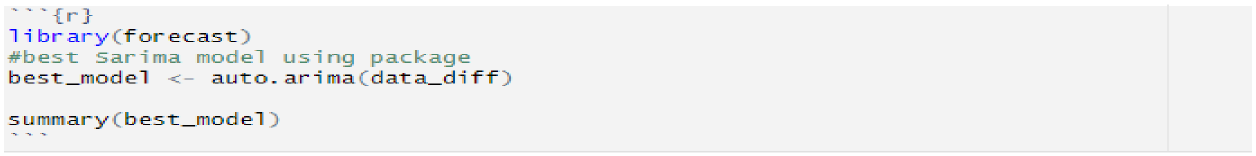
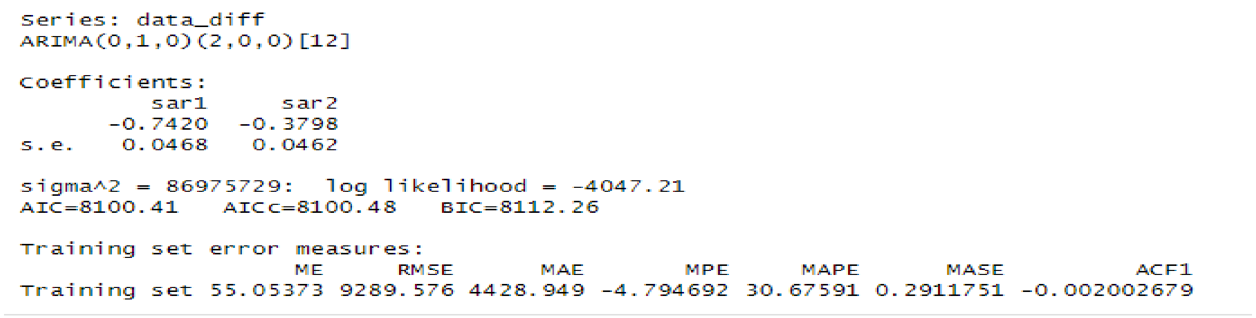
```
# Loading necessary libraries

library(forecast)

# Defining parameter grids

ps <- seq(from = 0, to = 2, by = 1)

qs <- seq(from = 0, to = 2, by = 1)

Ps <- seq(from = 0, to = 2, by = 1)
```

```
Qs <- seq(from = 0, to = 1, by = 1)

d = 1

D = 1

# Initializing results storage

results <- data.frame(AIC = numeric(),

                      p = integer(),

                      q = integer(),

                      P = integer(),

                      Q = integer(),

                      SSE = numeric(),

                      p_value = numeric())

# Loop through parameter combinations

for (p in ps) {

  for (q in qs) {

    for (P in Ps) {

      for (Q in Qs) {

        # Ensuring the sum of orders is within a reasonable limit

        if (p + q +d +D + P + Q <= 10) {

          # Fitting SARIMA model

          model <- tryCatch(

            arima(x = data_diff,

                  order = c(p, d, q),

                  seasonal = list(order = c(P, D, Q), period = 12)),

            error = function(e) NULL
          )


          # Checking if model fitting was successful
          if (!is.null(model)) {
            # Calculate sum of squared errors
            sse <- sum(model$residuals^2)


            # Perform Ljung-Box test for residual autocorrelation
            ljung_box_test <- Box.test(model$residuals, lag =
log(length(model$residuals)))
            p_value <- ljung_box_test$p.value


            # Store results
            results <- rbind(results, c(model$aic, p, q, P, Q, sse, p_value))
          }
        }
      }
    }
  }
}
# Name columns
colnames(results) <- c("AIC", "p", "q", "P", "Q", "SSE", "p_value")
# Sort results by AIC in ascending order
results <- results[order(results$AIC), ]
results
```


Description: df [52 × 7]

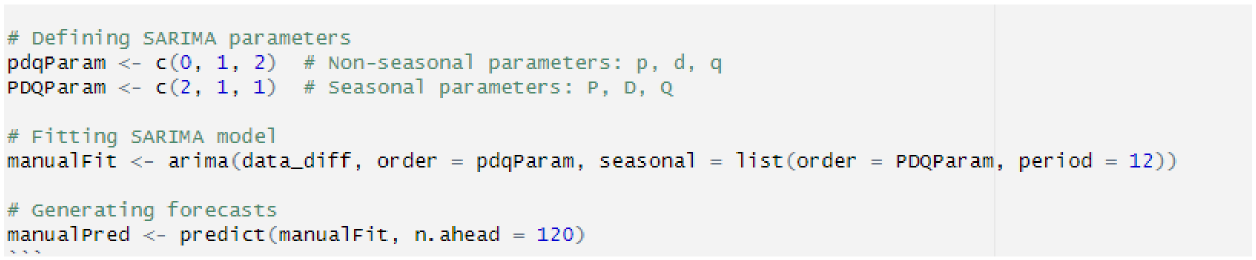
	AIC <dbl>	p <dbl>	q <dbl>	P <dbl>	Q <dbl>	SSE <dbl>	p_value <dbl>
18	7900.293	0	2	2	1	31303008656	8.222315e-01
34	7902.429	1	2	2	1	31385978901	7.775964e-01
52	7903.719	2	2	2	1	31255855693	9.598288e-01
40	7905.260	2	0	2	1	31704486179	2.064977e-01
46	7907.234	2	1	2	1	31693275634	1.866798e-01
28	7917.048	1	1	2	1	32673876470	1.973586e-02
6	7919.579	0	0	2	1	33192049074	9.848075e-04
12	7921.579	0	1	2	1	33192496012	9.869268e-04
16	7941.176	0	2	1	1	36096654781	6.908037e-01
32	7941.680	1	2	1	1	35954593934	9.694179e-01

As an alternative option used package to determine the best model


~~~
```{r}
library(forecast)
#best Sarima model using package
best_model  <- auto.arima(data_diff)

summary(best_model)
~~~
```


```
Series: data_diff 
ARIMA(0,1,0)(2,0,0)[12]

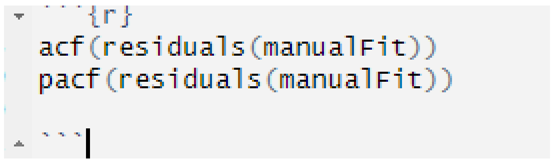
Coefficients:
         sar1     sar2
      -0.7420  -0.3798
s.e.   0.0468   0.0462

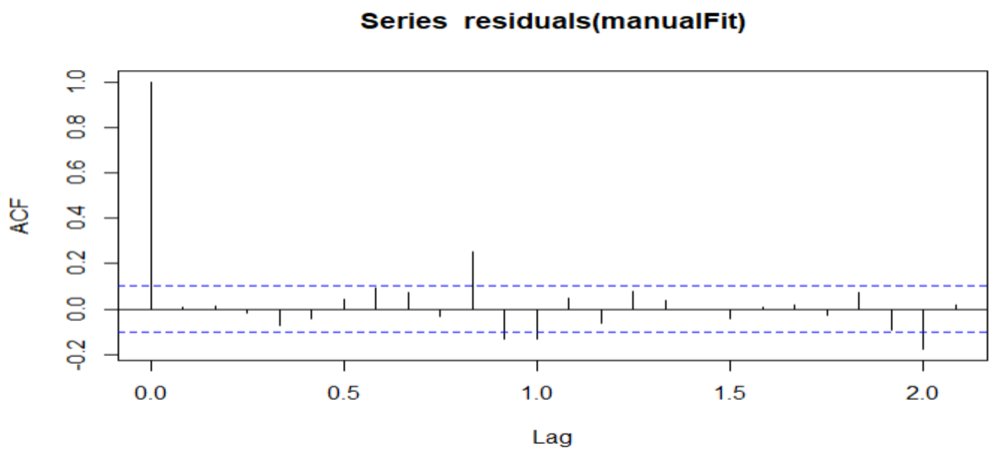
sigma^2 = 86975729:  log likelihood = -4047.21
AIC=8100.41   AICc=8100.48   BIC=8112.26

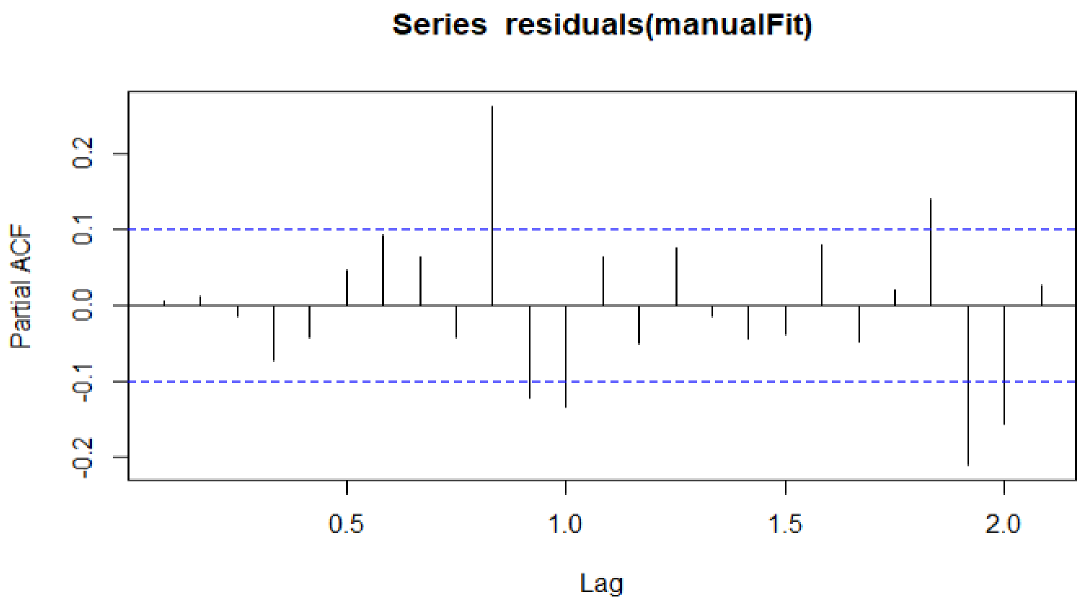
Training set error measures:
                   ME     RMSE      MAE       MPE     MAPE      MASE         ACF1
Training set 55.05373 9289.576 4428.949 -4.794692 30.67591 0.2911751 -0.002002679
```

Comparing the AIC and BIC values, we concluded the best model as ARIMA(0,1,2)(2,1,1)

Fitting a SARIMA Model:


```
# Defining SARIMA parameters
pdqParam <- c(0, 1, 2)  # Non-seasonal parameters: p, d, q
PDQParam <- c(2, 1, 1)  # Seasonal parameters: P, D, Q

# Fitting SARIMA model
manualFit <- arima(data_diff, order = pdqParam, seasonal = list(order = PDQParam, period = 12))

# Generating forecasts
manualPred <- predict(manualFit, n.ahead = 120)
```
```


~~~
```{r}
acf(residuals(manualFit))
pacf(residuals(manualFit))

```
~~~

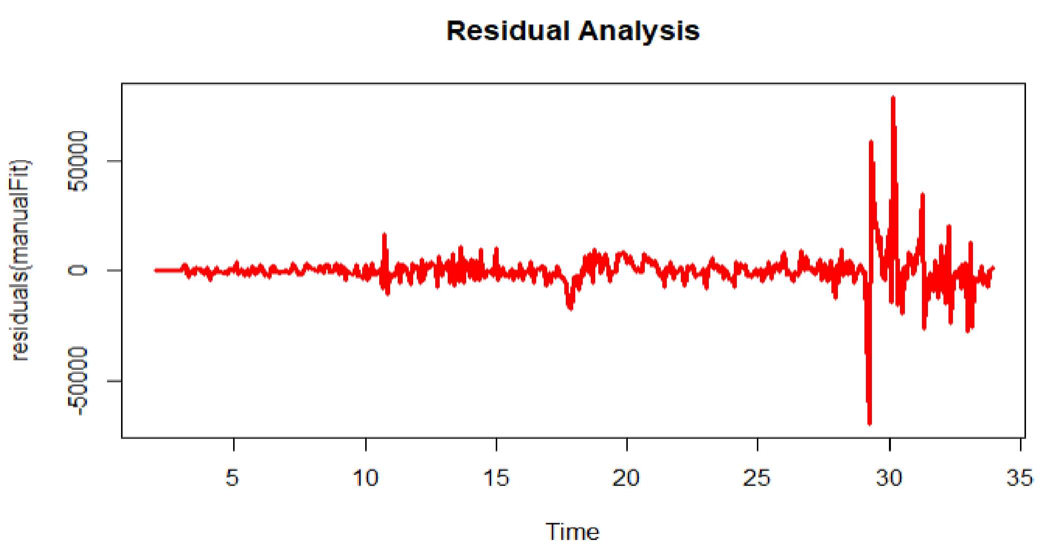




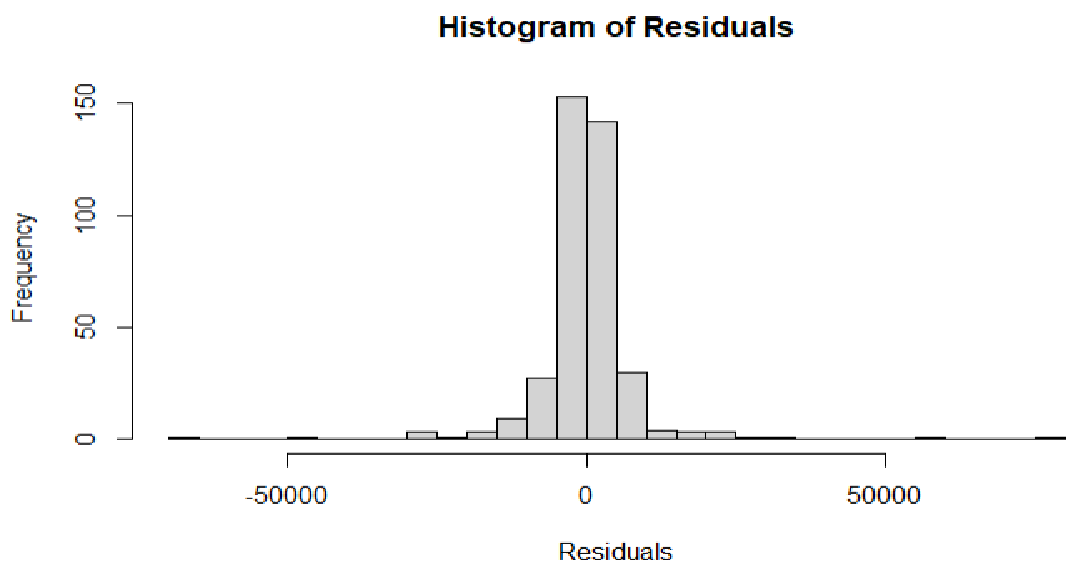
Residual Analysis:

Plotting the autocorrelations of the residuals of the Forecasted series.

```{r}
ts.plot(residuals(manualFit),lwd=3,col="red",main='Residual Analysis')
```

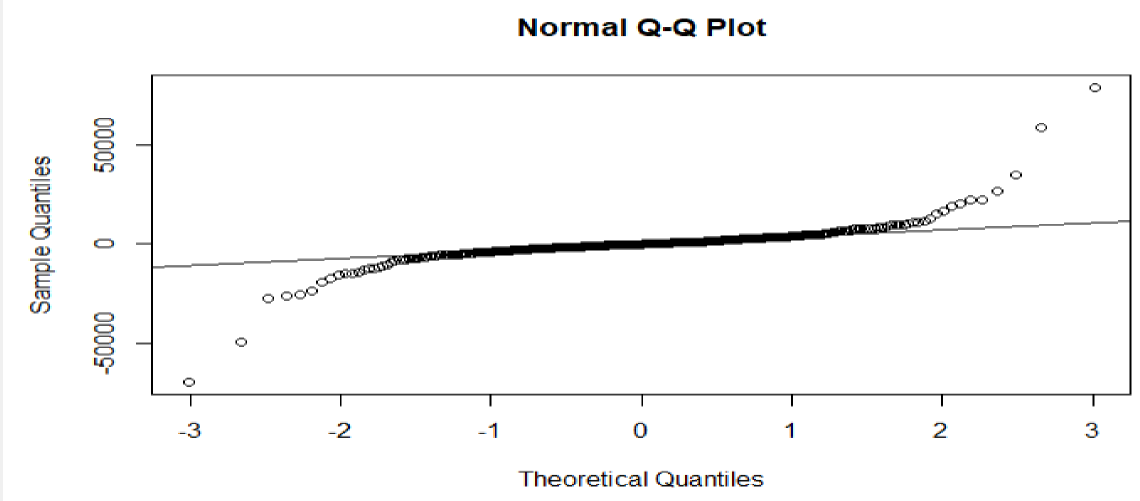


```{r}
hist(residuals(manualFit), main="Histogram of Residuals", xlab="Residuals", breaks=30)
```



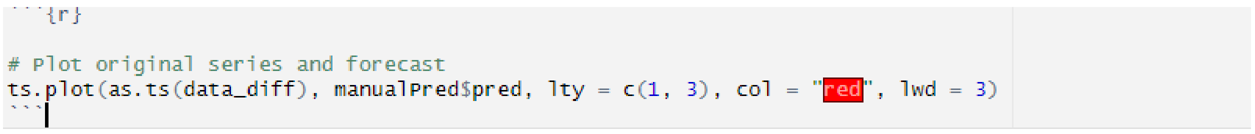
The histogram looks relatively symmetric, but the range of the residuals shows slight skewness in the distribution. This could be further confirmed with statistical tests or Q-Q plots.

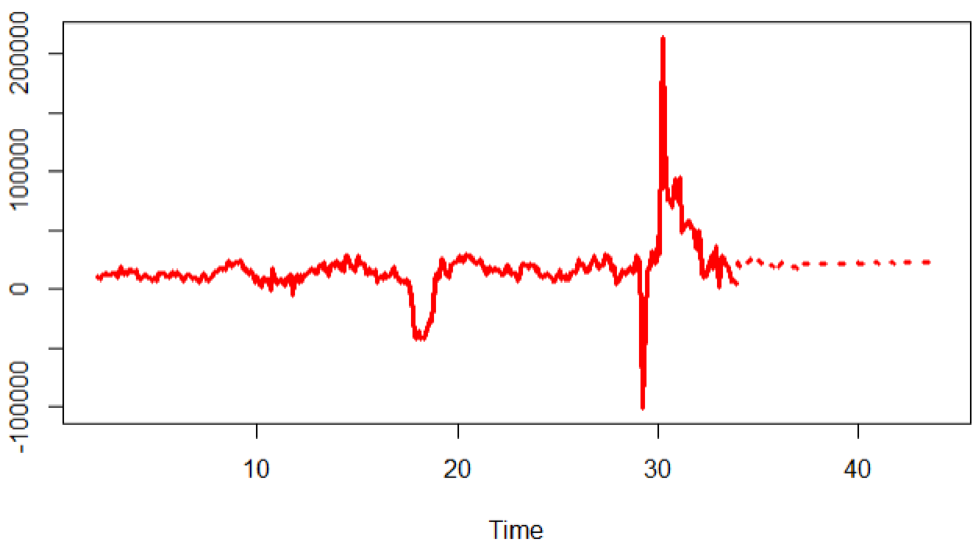
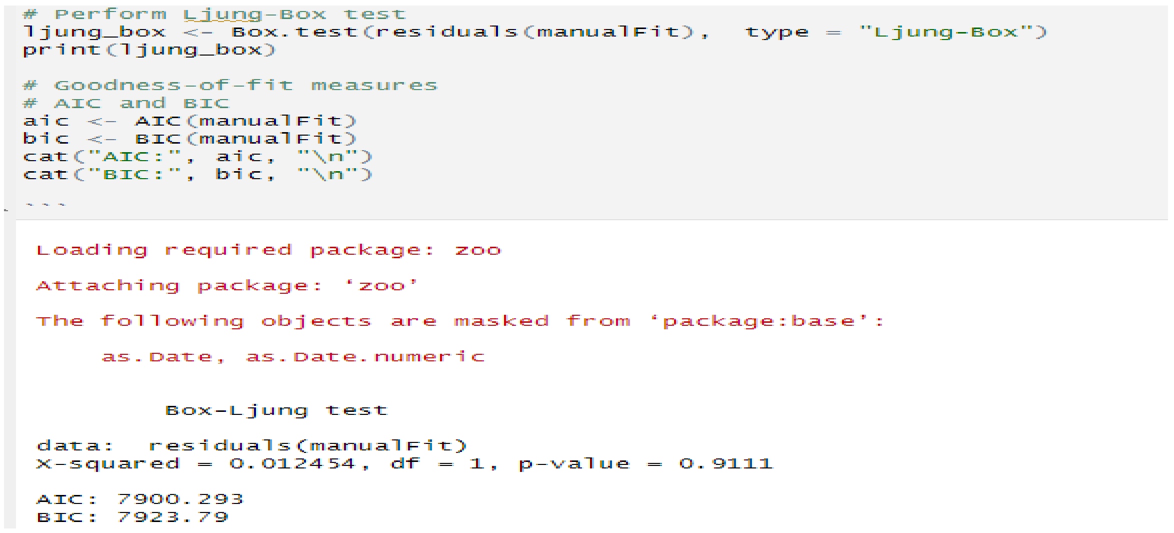
```
qqnorm(residuals(manualFit))
qqline(residuals(manualFit))
```



The central points lie along the straight diagonal line, indicating that most of the residuals closely follow a normal distribution around the mean.The points at both ends (the tails) deviate significantly from the diagonal line, suggesting that the residuals have heavier tails than the normal distribution.

Forecasting:



~~~
```{r}

# Plot original series and forecast
ts.plot(as.ts(data_diff), manualPred$pred, lty = c(1, 3), col = "red", lwd = 3)
~~~
```



200000
100000
0
-100000
10
20
30
40
Time


Plotted the time series of the prediction with respect to the original data. Comparing with the original data, we see that the forecast assumes improved future performance.



```
Perform Ljung-Box test
ljung_box <- Box.test(residuals(manualFit), type = "Ljung-Box")
print(ljung_box)

Goodness-of-fit measures
AIC and BIC
aic <- AIC(manualFit)
bic <- BIC(manualFit)
cat("AIC:", aic, "\n")
cat("BIC:", bic, "\n")

```
```

```
Loading required package: zoo

Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric


        Box-Ljung test

data:  residuals(manualFit)
X-squared = 0.012454, df = 1, p-value = 0.9111

AIC: 7900.293
BIC: 7923.79
```

Conclusion:

As the p value is greater than 0.05 then the residuals are independent which we want for the model to be correct. The p-value of 0.9111 is high, indicating that we cannot reject the null hypothesis. This means that there isn't significant autocorrelation in the residuals of the model, and the model fits the data well.

Non-Seasonal: Time series analysis for PAYEMS

Abstract :

The study analyzes Payroll Employment (PAYEMS) data, offering insights into long-term employment trends without the influence of seasonal factors. Utilizing advanced statistical techniques, the research examines the underlying patterns of PAYEMS, revealing key economic indicators and trends. This non-seasonal analysis aims to identify structural changes, growth patterns, and significant fluctuations in employment across various sectors. The findings provide policymakers and economists with a comprehensive understanding of long-term employment dynamics, enhancing strategic decision-making.

Introduction:

Employment trends serve as a critical barometer for economic health, providing insights into consumer spending, economic stability, and productivity. Payroll Employment (PAYEMS) data, published monthly by the U.S. Bureau of Labor Statistics, is pivotal in identifying workforce trends. However, its seasonal nature often requires filtering to reveal core trends. This study focuses on non-seasonal time series analysis of PAYEMS data to identify persistent trends, structural changes, and macroeconomic shifts affecting employment. By excluding seasonal variations, the analysis uncovers long-term trends vital for policy formulation, economic forecasting, and strategic business planning.

The methodology incorporates statistical smoothing techniques to identify patterns that reflect broader economic conditions. This study aims to contribute to the growing body of research by offering a clear view of employment trends, helping stakeholders make informed decisions for sustainable economic growth.

Data Description:

Date Range: From January 1939 – March 2024

Dataset Description: The dataset contains historical payroll employment (PAYEMS) data for the United States, starting from January 1939. This data is presented monthly, reflecting the total number of employees on non-farm payrolls across the nation. Each data point includes the corresponding date and the payroll employment figure for that month.

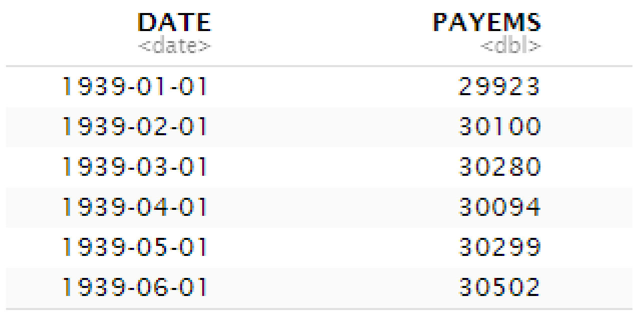
Datasource Description: This data is form FRED Economic data website and can be accessed by using link: https://fred.stlouisfed.org/series/PAYEMS

DATE: Monthly timestamp indicating the date for the specific PAYEMS data point, formatted as YYYY-MM-DD.

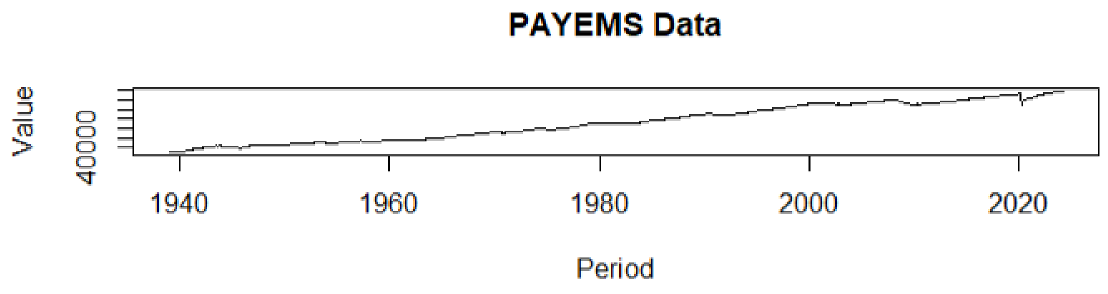
PAYEMS: Total number of employees (in thousands) on non-farm payrolls in the U.S.

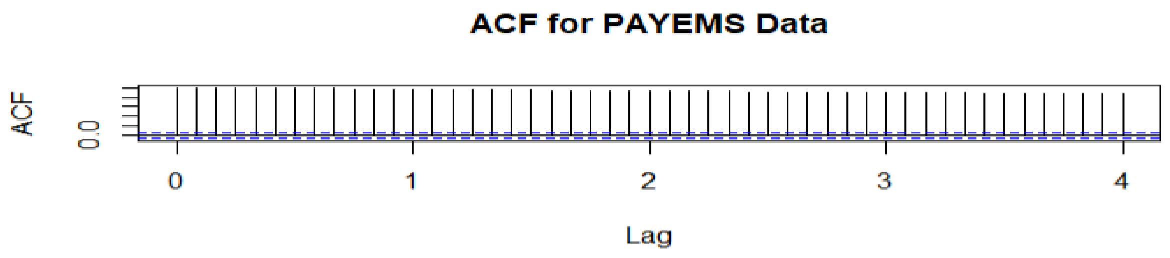
Initial look at the data:

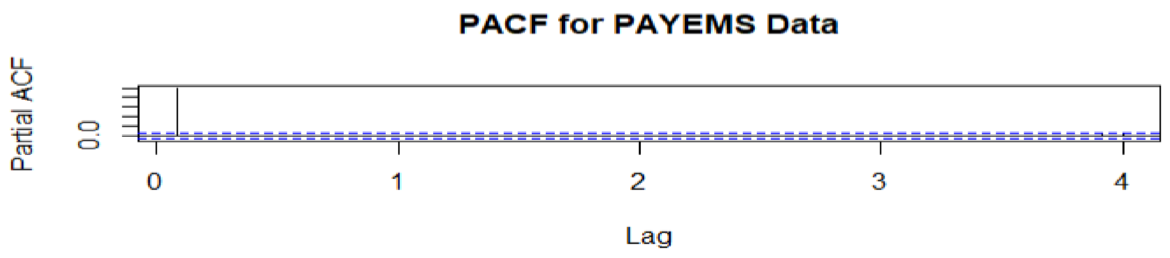
Each data point is associated with a specific date and the corresponding monthly payroll employment figure, representing total non-farm payroll employment in the U.S. The data is straightforward, featuring only two columns:



DATE <date>	PAYEMS <dbl>
1939-01-01	29923
1939-02-01	30100
1939-03-01	30280
1939-04-01	30094
1939-05-01	30299
1939-06-01	30502





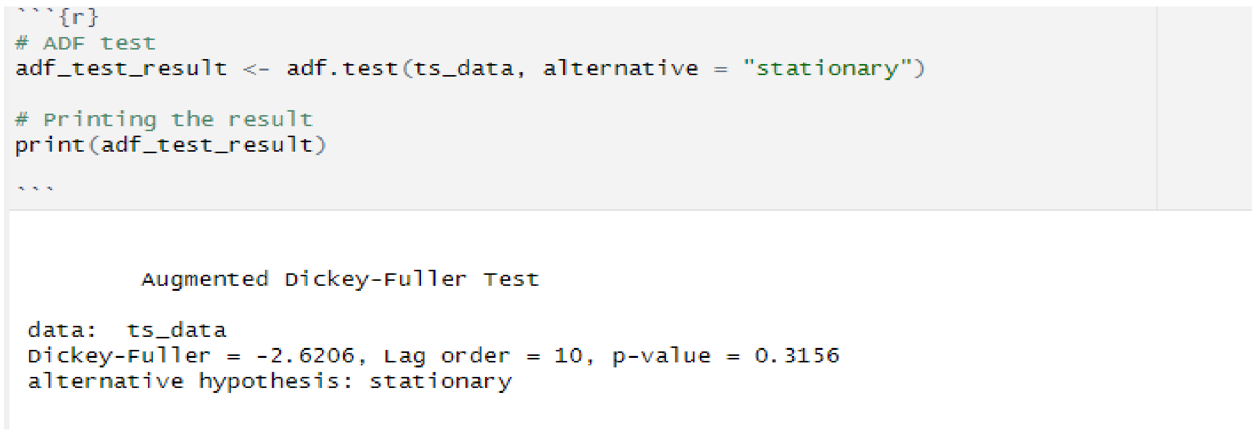


The plot clearly shows a steady upward trend in payroll employment, indicating long-term growth in non-farm employment. This is consistent with the general expansion of the U.S. economy over time.

FITTING THE TIME SERIES MODEL

Stationarity Check :

Let's start by checking if the time series is stationary or not. To do so we are going to use the Dickey Fuller and/or augmented Dickey fuller test.

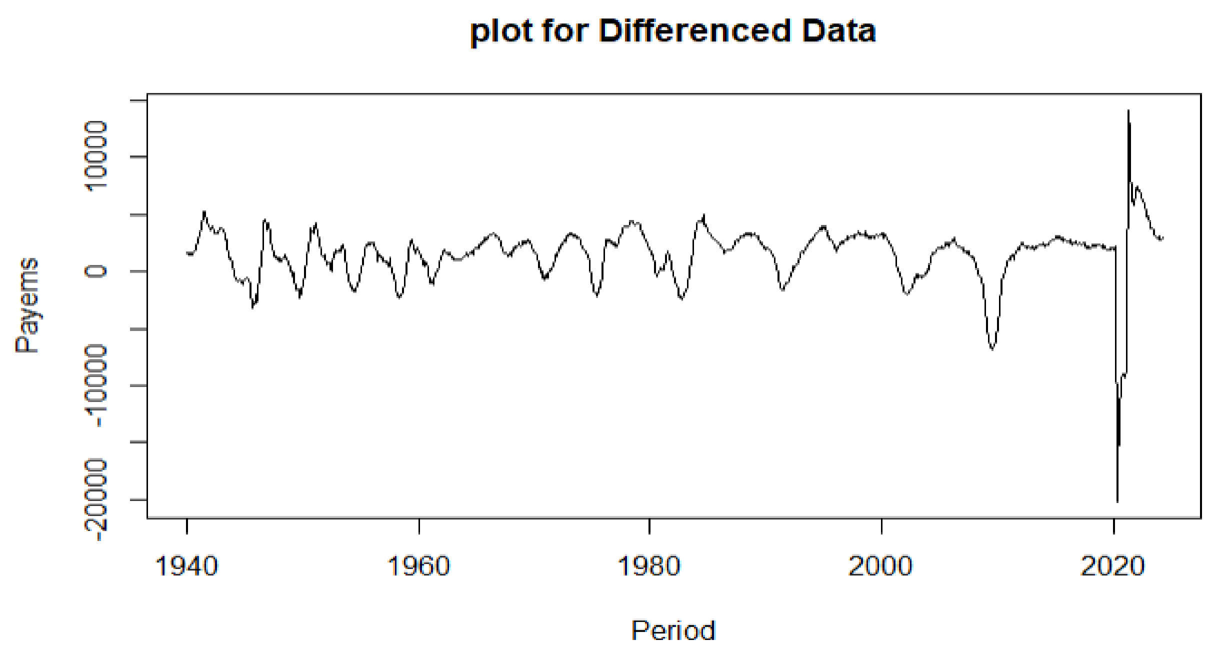


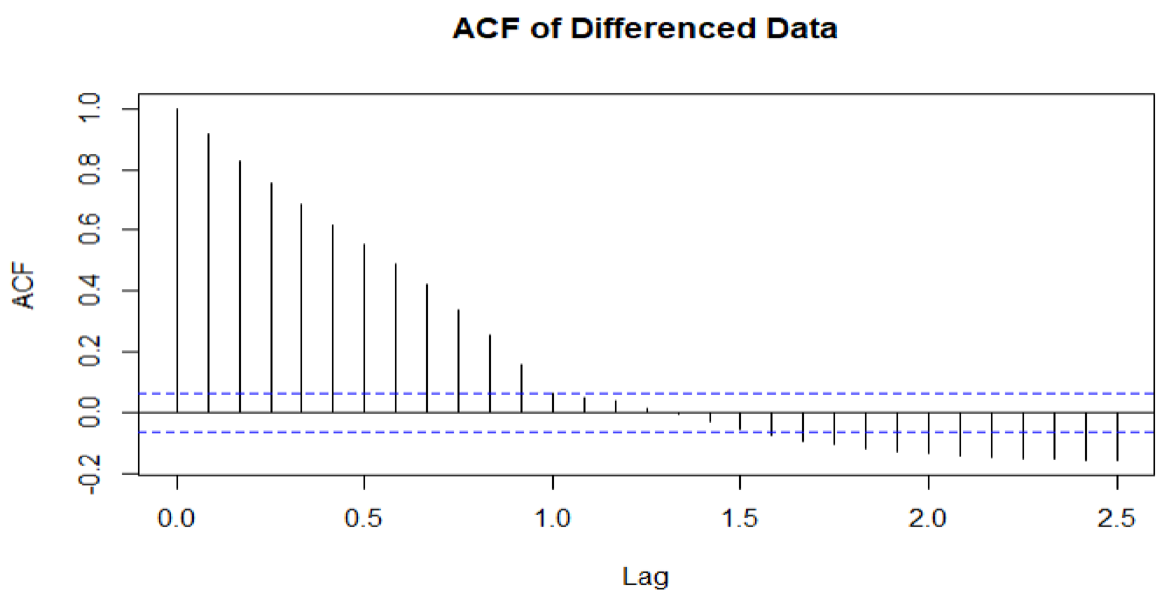
~~~
```{r}
# ADF test
adf_test_result <- adf.test(ts_data, alternative = "stationary")

# Printing the result
print(adf_test_result)

```
~~~

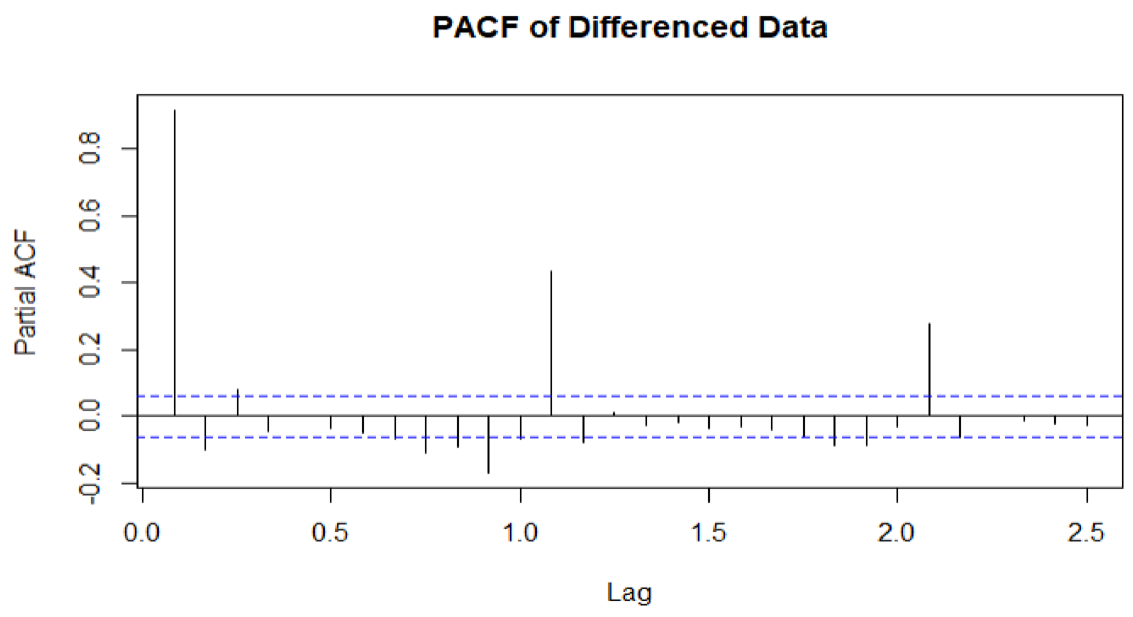
```

	Augmented Dickey-Fuller Test

data:  ts_data
Dickey-Fuller = -2.6206, Lag order = 10, p-value = 0.3156
alternative hypothesis: stationary
```

Since the p-value is greater than 0.05 indicating non-stationarity, differencing the data and checking again for stationarity.

```
ts_data_diff <- diff(ts_data, lag = 12)

# Now perform the ADF test on the differenced data
adf_test_result <- adf.test(ts_data_diff, alternative="stationary")

# Print the result
print(adf_test_result)

```
```

```
Warning: p-value smaller than printed p-value
        Augmented Dickey-Fuller Test

data:  ts_data_diff
Dickey-Fuller = -9.2378, Lag order = 10, p-value = 0.01
alternative hypothesis: stationary
```

Now the p-value is less than 0.05 indicating it stationary

It is important to make the series stationary before fitting a model to it, because, we observe a single run of a stochastic process rather than repeated runs of the stochastic process. Hence, we need stationarity and ergodicity so that observing a long run of a stochastic process is similar to observing many independent runs of a stochastic process.

Plotting the Differenced data, ACF and PACF plots :


```

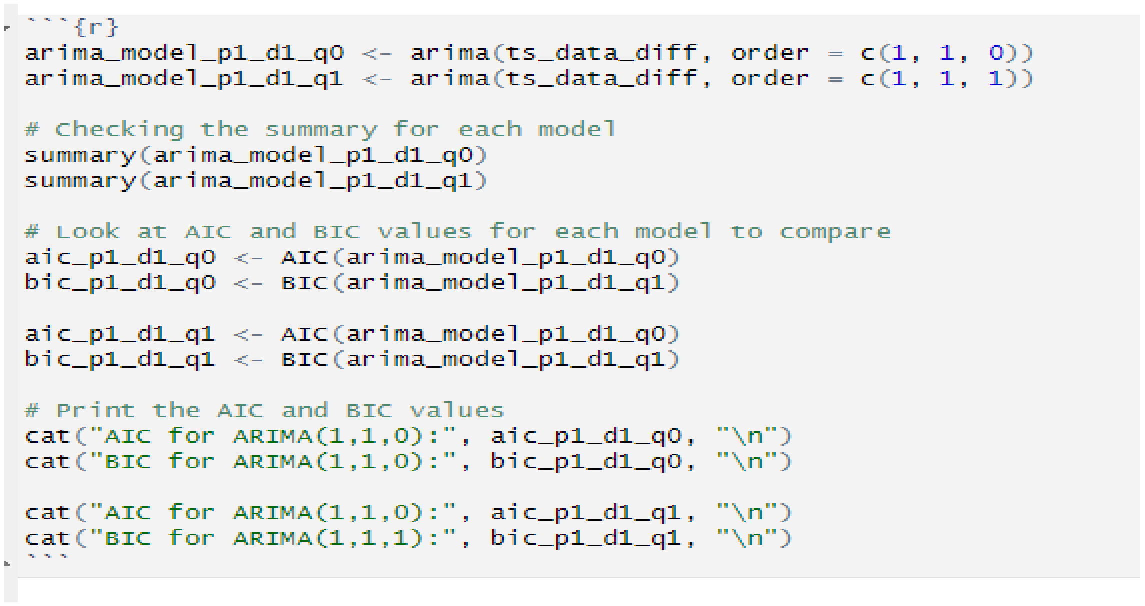




DETERMINING THE ORDER OF THE MODEL:

As we know that the ARMA models are used to capture the constant mean in the series, and the constant variance will be captured by the GARCH model. We can see from the return plots that there is a constant mean. Then we will determine the order of the GARCH model by using the ACF and PACF plots. As we are using the ugrachfit method from the "rugarch" library instead of the garch method from the “tseries” library. We directly give the ARMA and GRACH order to the method ugrachfit and the return data instead of giving the residuals as we would have for the garch method.

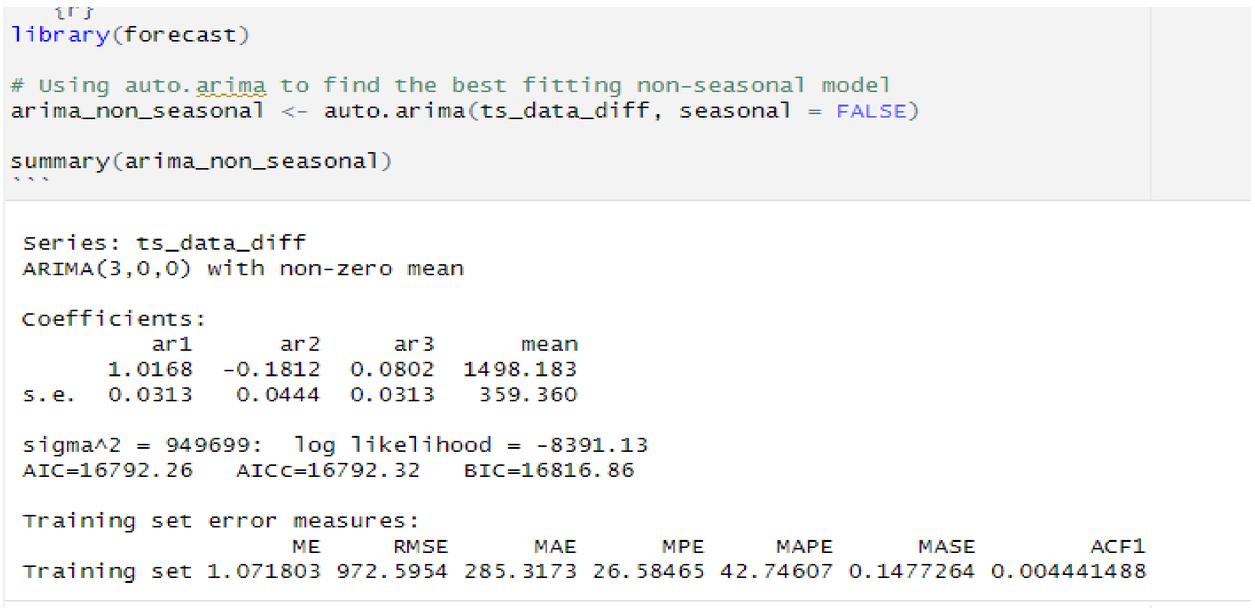
It looks like multiple AR terms could be included, but further testing is needed to determine the exact value. Start with p = 1 or p = 2. Similarly, the MA order could require a higher value. Starting with q = 1 or q = 2 is recommended.



~~~
```{r}
arima_model_p1_d1_q0 <- arima(ts_data_diff, order = c(1, 1, 0))
arima_model_p1_d1_q1 <- arima(ts_data_diff, order = c(1, 1, 1))

# Checking the summary for each model
summary(arima_model_p1_d1_q0)
summary(arima_model_p1_d1_q1)

# Look at AIC and BIC values for each model to compare
aic_p1_d1_q0 <- AIC(arima_model_p1_d1_q0)
bic_p1_d1_q0 <- BIC(arima_model_p1_d1_q1)

aic_p1_d1_q1 <- AIC(arima_model_p1_d1_q0)
bic_p1_d1_q1 <- BIC(arima_model_p1_d1_q1)

# Print the AIC and BIC values
cat("AIC for ARIMA(1,1,0):", aic_p1_d1_q0, "\n")
cat("BIC for ARIMA(1,1,0):", bic_p1_d1_q0, "\n")

cat("AIC for ARIMA(1,1,0):", aic_p1_d1_q1, "\n")
cat("BIC for ARIMA(1,1,1):", bic_p1_d1_q1, "\n")
```
~~~

```
Call:
arima(x = ts_data_diff, order = c(1, 1, 0))

Coefficients:
         ar1
      0.0549
s.e.  0.0314

sigma^2 estimated as 1000977:  log likelihood = -8410.46,  aic = 16824.91

Training set error measures:
                   ME     RMSE      MAE      MPE     MAPE      MASE        ACF1
Training set 1.175965 999.9933 268.7851 21.32195 41.96573 0.9844002 0.006899323

Call:
arima(x = ts_data_diff, order = c(1, 1, 1))

Coefficients:
          ar1     ma1
      -0.8826  0.9982
s.e.   0.0157  0.0052

sigma^2 estimated as 946251:  log likelihood = -8383.48,  aic = 16772.97

Training set error measures:
                   ME    RMSE      MAE      MPE     MAPE     MASE         ACF1
Training set 1.141171 972.273 277.5756 19.81215 40.54396 1.016594 -0.002959577
AIC for ARIMA(1,1,0): 16824.91
BIC for ARIMA(1,1,0): 16787.72
AIC for ARIMA(1,1,0): 16824.91
BIC for ARIMA(1,1,1): 16787.72
```

As an alternative option used package to determine the best model



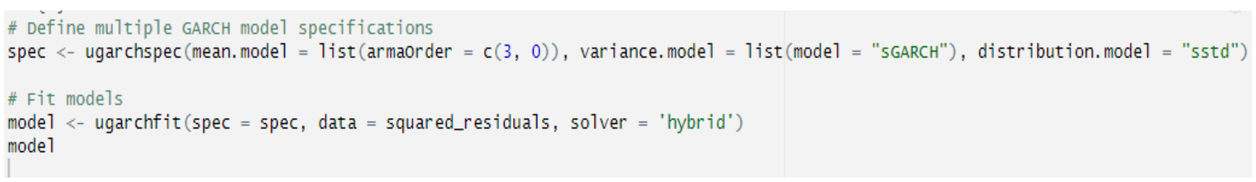
~~~
{r}
library(forecast)

# Using auto.arima to find the best fitting non-seasonal model
arima_non_seasonal <- auto.arima(ts_data_diff, seasonal = FALSE)

summary(arima_non_seasonal)
```
~~~

```
Series: ts_data_diff
ARIMA(3,0,0) with non-zero mean

Coefficients:
         ar1      ar2     ar3      mean
      1.0168  -0.1812  0.0802  1498.183
s.e.  0.0313   0.0444  0.0313   359.360

sigma^2 = 949699:  log likelihood = -8391.13
AIC=16792.26   AICc=16792.32   BIC=16816.86

Training set error measures:
                   ME     RMSE      MAE      MPE     MAPE      MASE        ACF1
Training set 1.071803 972.5954 285.3173 26.58465 42.74607 0.1477264 0.004441488
```

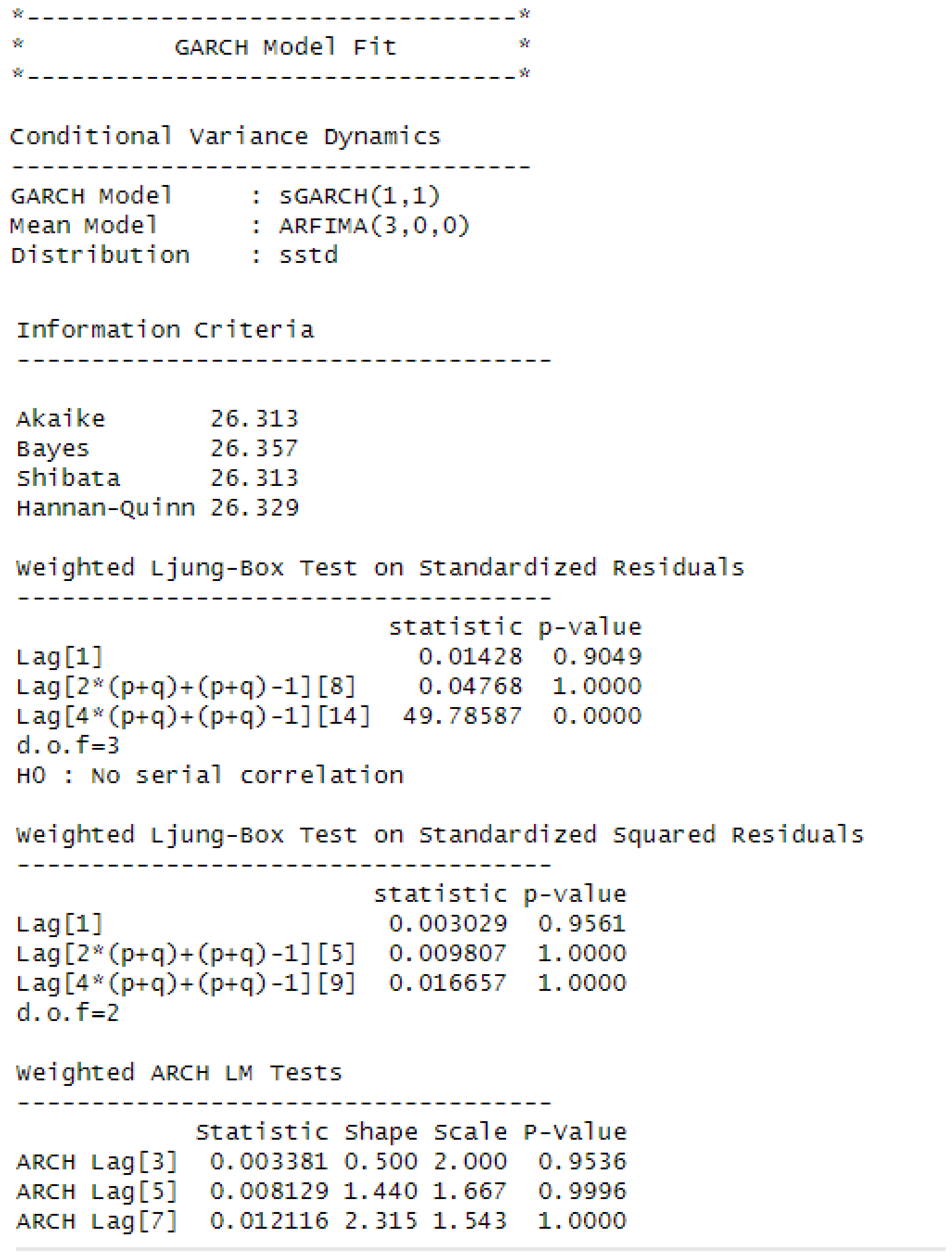
Comparing the AIC and BIC values, we concluded the best model as ARIMA(3,0,0)

Finding the best GARCH Model:

Skewed student t distribution GARCH model

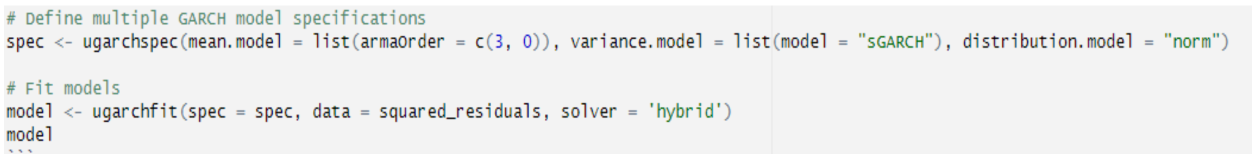


```
# Define multiple GARCH model specifications
spec <- ugarchspec(mean.model = list(armaOrder = c(3, 0)), variance.model = list(model = "sGARCH"), distribution.model = "sstd")

# Fit models
model <- ugarchfit(spec = spec, data = squared_residuals, solver = 'hybrid')
model
```

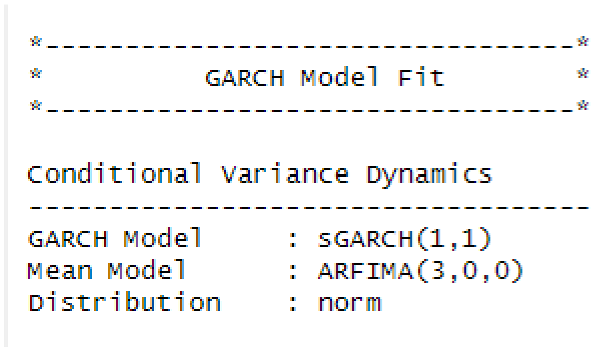


```
*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*

Conditional Variance Dynamics
-----------------------------------
GARCH Model      : sGARCH(1,1)
Mean Model       : ARFIMA(3,0,0)
Distribution     : sstd

 Information Criteria
 ------------------------------------

 Akaike       26.313
 Bayes        26.357
 Shibata      26.313
 Hannan-Quinn 26.329

 Weighted Ljung-Box Test on Standardized Residuals
 ------------------------------------
                         statistic p-value
 Lag[1]                    0.01428  0.9049
 Lag[2*(p+q)+(p+q)-1][8]   0.04768  1.0000
 Lag[4*(p+q)+(p+q)-1][14] 49.78587  0.0000
 d.o.f=3
 H0 : No serial correlation

 Weighted Ljung-Box Test on Standardized Squared Residuals
 ------------------------------------
                        statistic p-value
 Lag[1]                  0.003029  0.9561
 Lag[2*(p+q)+(p+q)-1][5] 0.009807  1.0000
 Lag[4*(p+q)+(p+q)-1][9] 0.016657  1.0000
 d.o.f=2

 Weighted ARCH LM Tests
 ------------------------------------
             Statistic Shape Scale P-Value
 ARCH Lag[3]  0.003381 0.500 2.000  0.9536
 ARCH Lag[5]  0.008129 1.440 1.667  0.9996
 ARCH Lag[7]  0.012116 2.315 1.543  1.0000
```

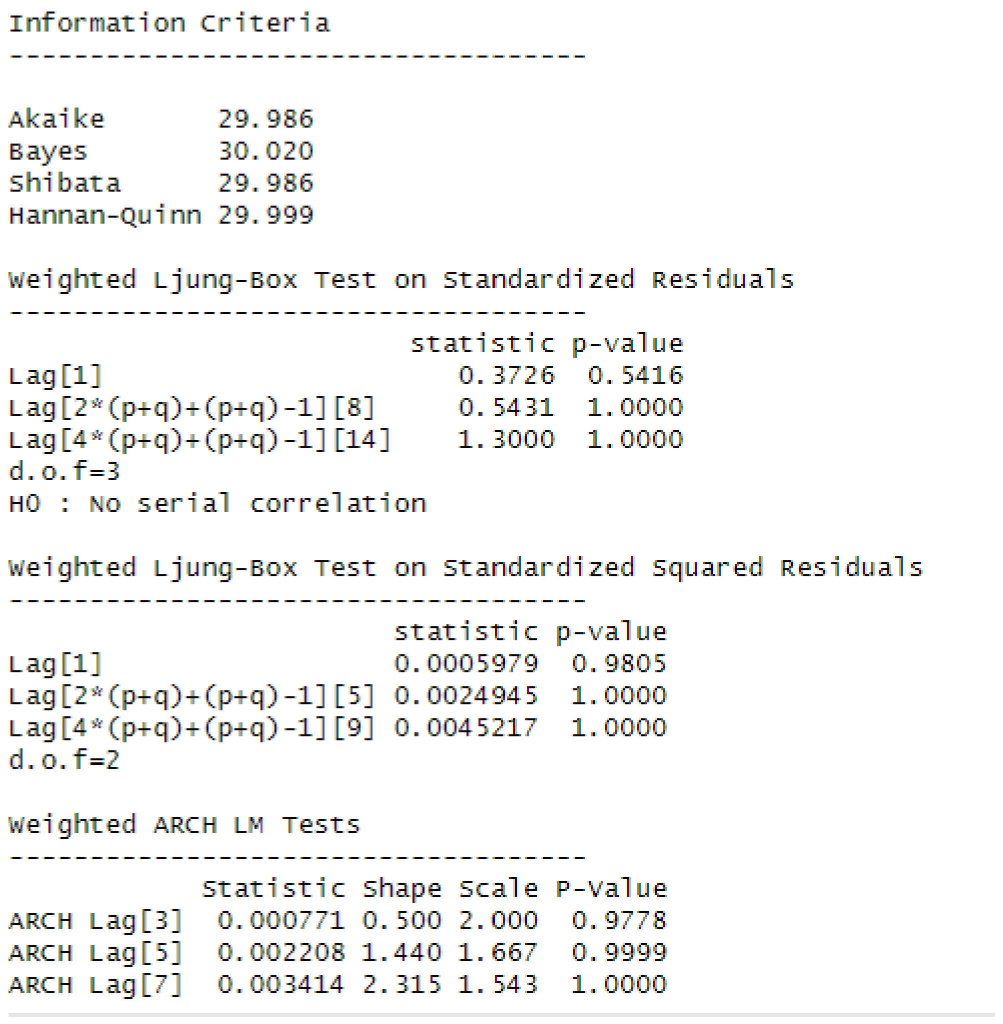
Normal distribution GARCH Model:



```r
# Define multiple GARCH model specifications
spec <- ugarchspec(mean.model = list(armaOrder = c(3, 0)), variance.model = list(model = "sGARCH"), distribution.model = "norm")

# Fit models
model <- ugarchfit(spec = spec, data = squared_residuals, solver = 'hybrid')
model
```



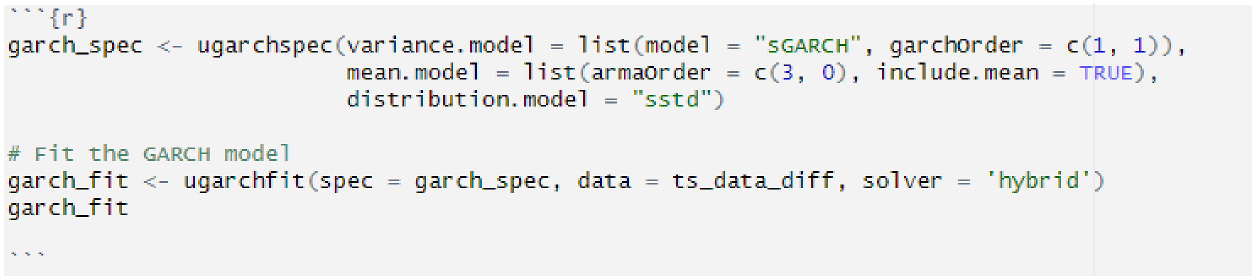
```
*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*

Conditional Variance Dynamics
-----------------------------------
GARCH Model     : sGARCH(1,1)
Mean Model      : ARFIMA(3,0,0)
Distribution    : norm
```



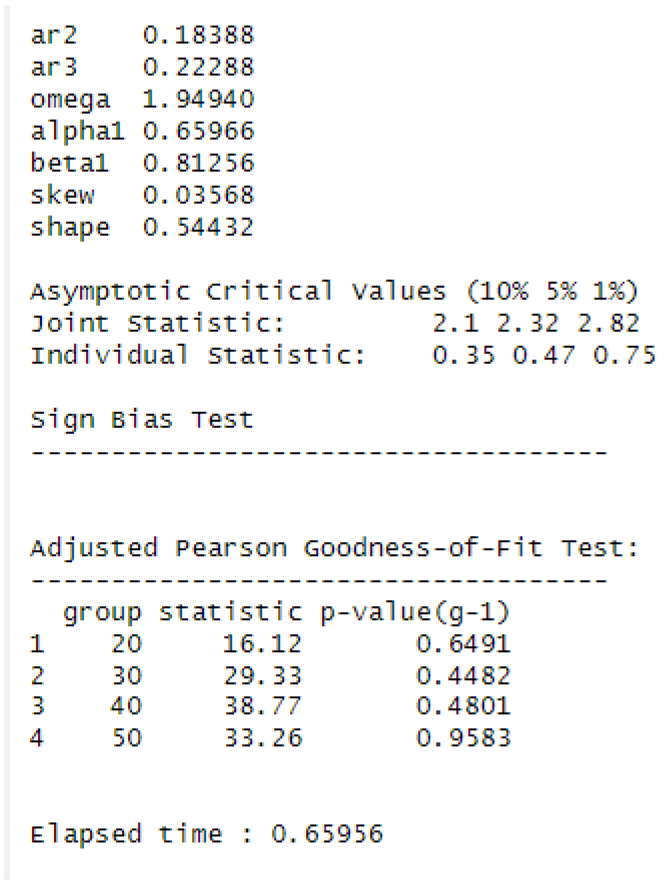
```
Information Criteria
------------------------------------

Akaike       29.986
Bayes        30.020
Shibata      29.986
Hannan-Quinn 29.999

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                         statistic p-value
Lag[1]                      0.3726  0.5416
Lag[2*(p+q)+(p+q)-1][8]     0.5431  1.0000
Lag[4*(p+q)+(p+q)-1][14]    1.3000  1.0000
d.o.f=3
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                  0.0005979  0.9805
Lag[2*(p+q)+(p+q)-1][5] 0.0024945  1.0000
Lag[4*(p+q)+(p+q)-1][9] 0.0045217  1.0000
d.o.f=2

Weighted ARCH LM Tests
------------------------------------
            Statistic Shape Scale P-Value
ARCH Lag[3]  0.000771 0.500 2.000  0.9778
ARCH Lag[5]  0.002208 1.440 1.667  0.9999
ARCH Lag[7]  0.003414 2.315 1.543  1.0000
```

Comparing AIC values of both GARCH models and taking GARCH with skewed student t distribution as it has lower AIC value

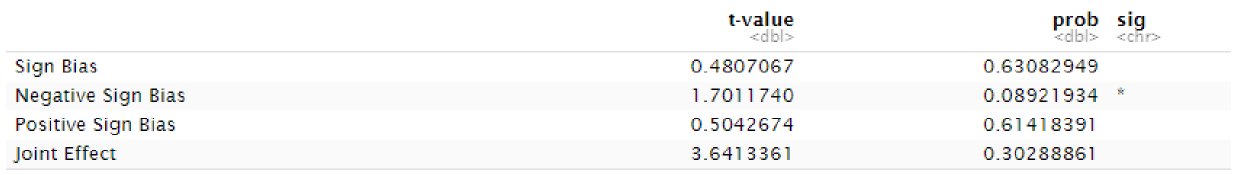
Fitting ARIMA-GARCH Model:



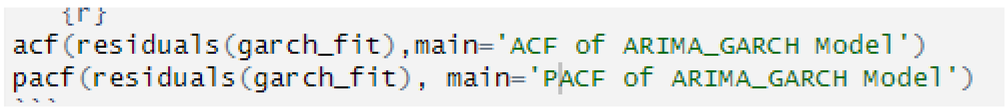
~~~
```{r}
garch_spec <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 1)),
                         mean.model = list(armaOrder = c(3, 0), include.mean = TRUE),
                         distribution.model = "sstd")

# Fit the GARCH model
garch_fit <- ugarchfit(spec = garch_spec, data = ts_data_diff, solver = 'hybrid')
garch_fit

```
~~~



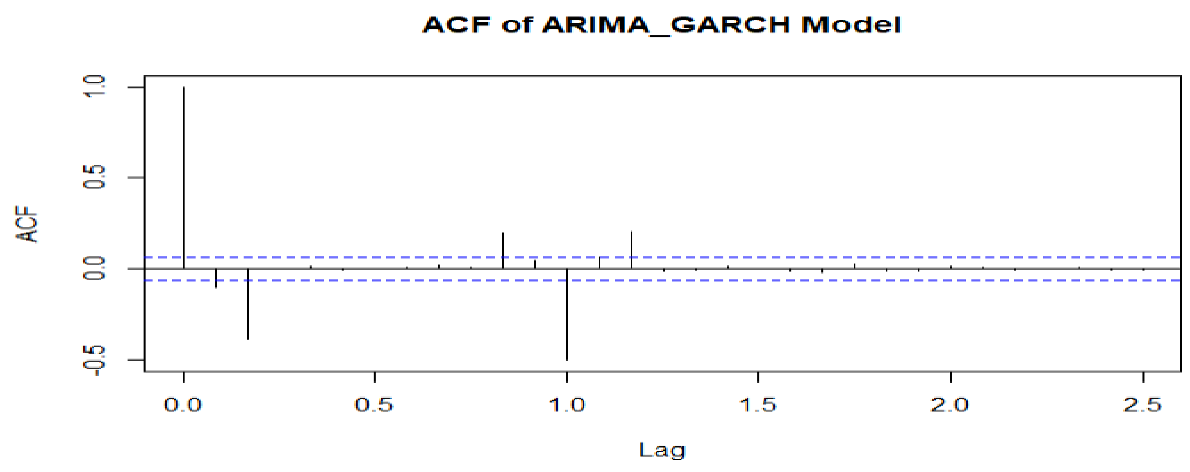
```
ar2     0.18388
ar3     0.22288
omega   1.94940
alpha1  0.65966
beta1   0.81256
skew    0.03568
shape   0.54432

Asymptotic Critical Values (10% 5% 1%)
Joint Statistic:        2.1 2.32 2.82
Individual Statistic:   0.35 0.47 0.75

Sign Bias Test
------------------------------------


Adjusted Pearson Goodness-of-Fit Test:
------------------------------------
  group statistic p-value(g-1)
1    20     16.12       0.6491
2    30     29.33       0.4482
3    40     38.77       0.4801
4    50     33.26       0.9583


Elapsed time : 0.65956
```

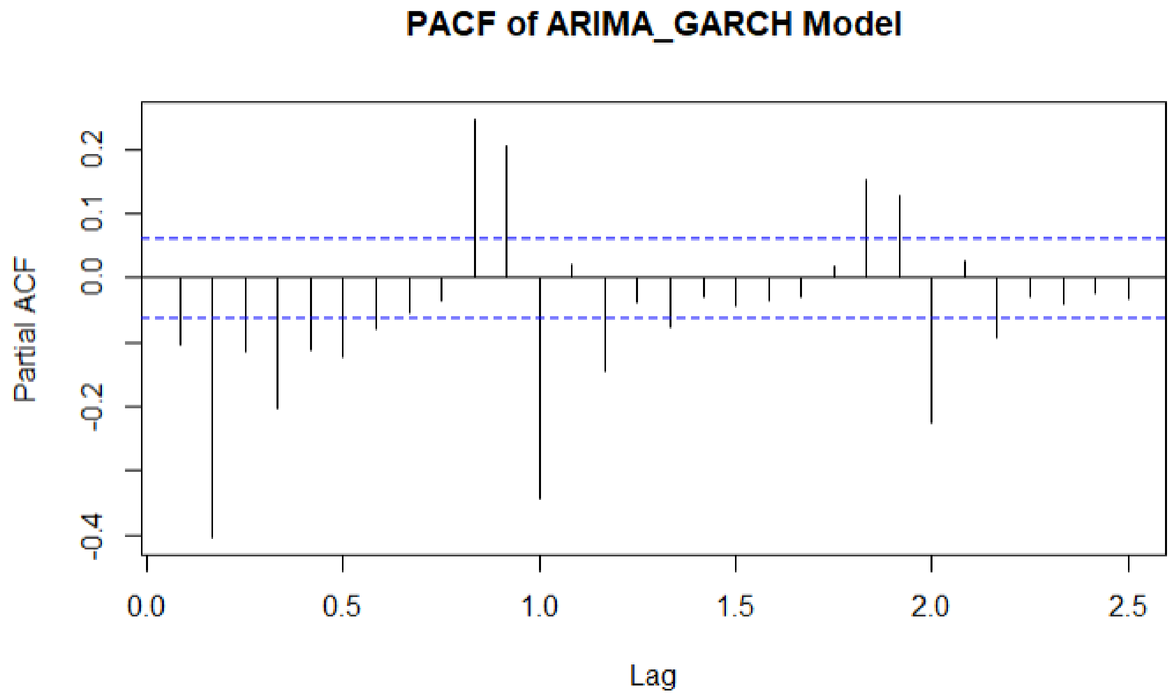


	t-value <dbl>	prob <dbl>	sig <chr>
Sign Bias	0.4807067	0.63082949	
Negative Sign Bias	1.7011740	0.08921934	*
Positive Sign Bias	0.5042674	0.61418391	
Joint Effect	3.6413361	0.30288861	



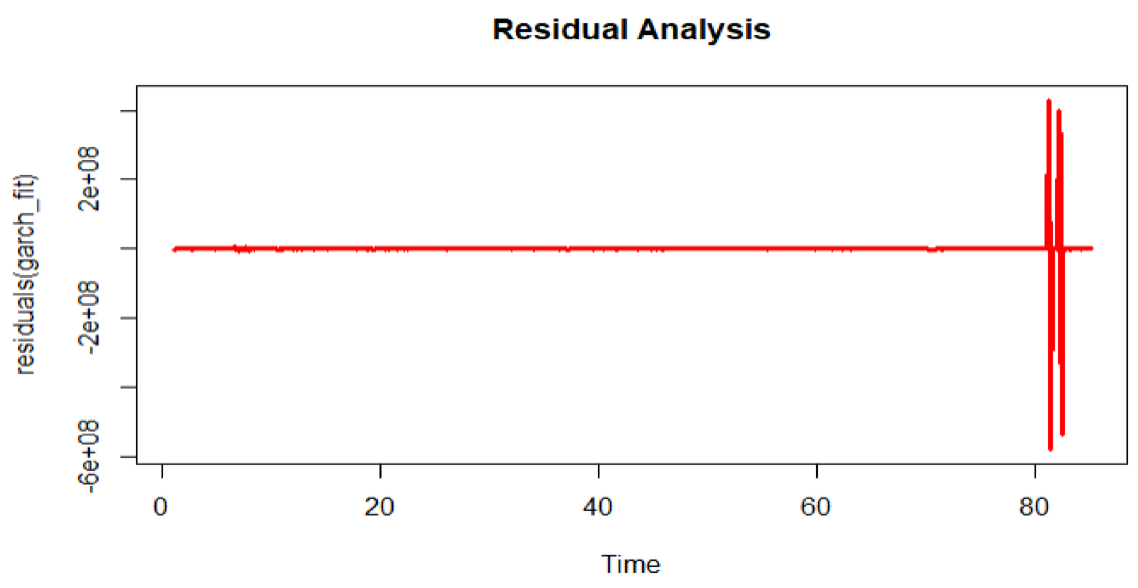
```
  {r}
acf(residuals(garch_fit),main='ACF of ARIMA_GARCH Model')
pacf(residuals(garch_fit), main='PACF of ARIMA_GARCH Model')
```



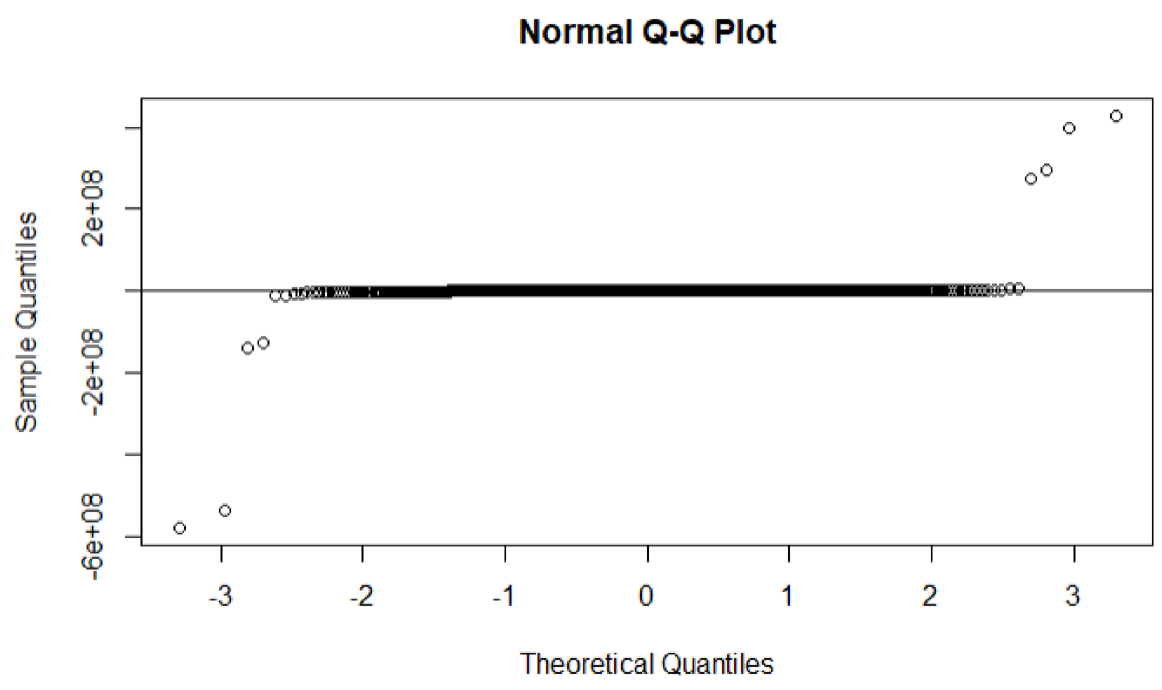


Residual analysis:

Plotting the autocorrelations of the residuals of the Forecasted series.

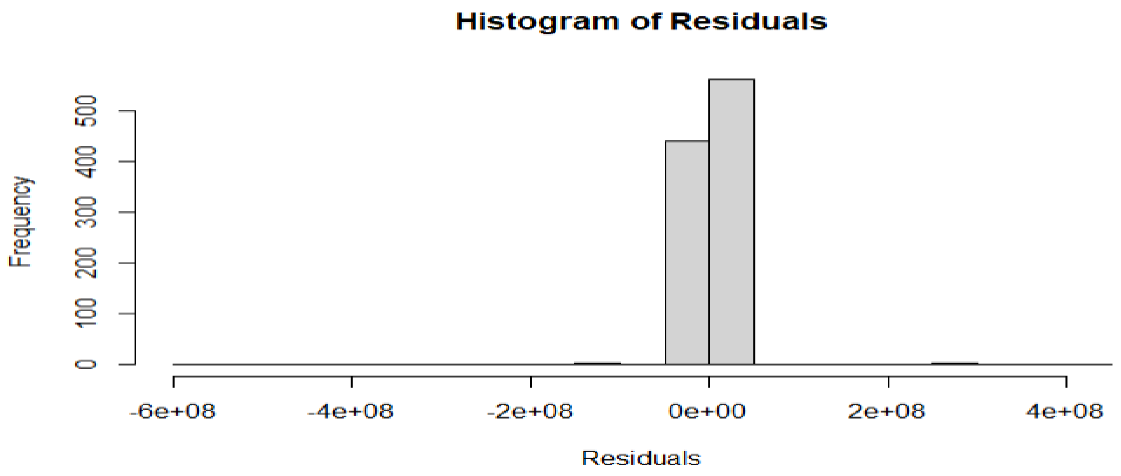


```
# Create a Q-Q plot of the residuals
qqnorm(residuals(garch_fit))
qqline(residuals(garch_fit))
```



The straight line represents a perfect normal distribution. Residuals falling closely along the line indicate they are normally distributed.

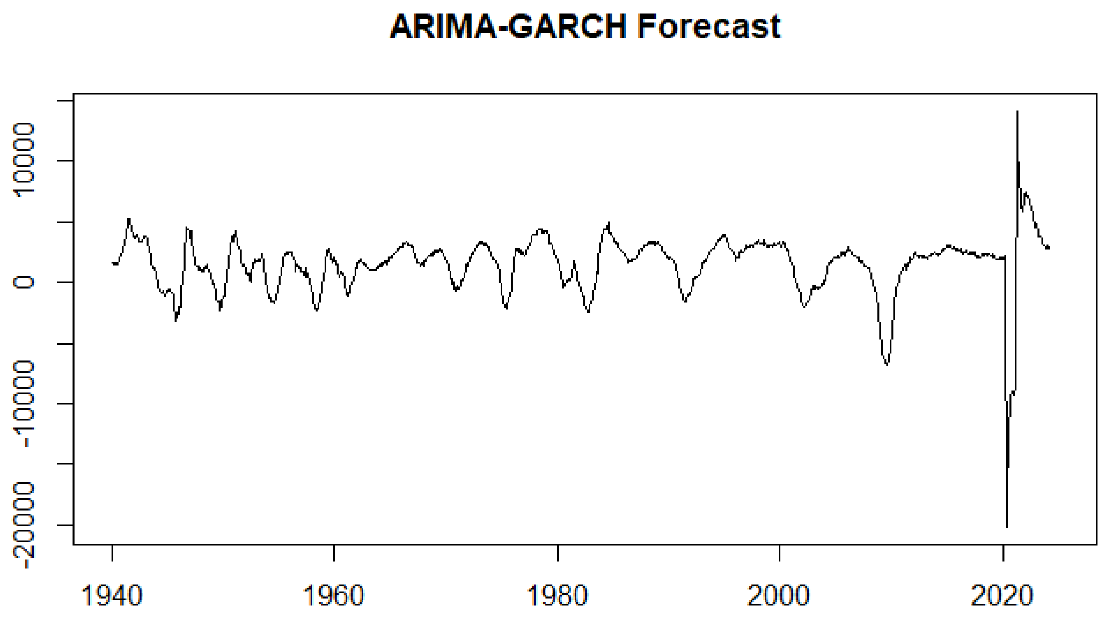
```{r}
hist(residuals(garch_fit), main="Histogram of Residuals", xlab="Residuals", breaks=30)
```



The histogram shows a skewed shape, implying asymmetry in the residuals. This is not uncommon in financial time series data where residuals may not always follow a symmetric distribution.

Forecasting:

```
# Forecast future values with ARIMA
arima_forecast <- forecast(arima_non_seasonal, h = 10)

# Forecast future volatility with GARCH
garch_forecast <- ugarchforecast(garch_fit, n.ahead = 10)

# Extract conditional standard deviations (volatility)
garch_volatility <- as.data.frame(sigma(garch_forecast))

# Combine forecasts
arima_forecast$lower <- arima_forecast$mean - 2 * garch_volatility[,1]
arima_forecast$upper <- arima_forecast$mean + 2 * garch_volatility[,1]
# Plot the ARIMA-GARCH forecast with prediction intervals
plot(arima_forecast, main = "ARIMA-GARCH Forecast")
```



Conclusion:

The graph indicates significant volatility at the forecast section, a characteristic captured well by the GARCH component. The GARCH model handles the clustering of volatility and the significant changes, especially during recent financial disruptions. The model captures the past data well and provides plausible forecasts based on recent historical volatility. Residual diagnostics would be useful for assessing if the model has any remaining patterns or if additional GARCH variants should be considered.

References:

- R Packages and Statistical Tools:
- Hyndman, R. J., & Athanasopoulos, G. (2018): Forecasting: Principles and Practice. Ghalanos, A. (2019): "rugarch: An R Package for Univariate GARCH ModelsKey
- Research Papers and Articles:Engle, R. F. (1982): "Autoregressive Conditional Heteroskedasticity with Estimates of the Variance of United Kingdom Inflation." Econometrica, 50(4), 987-1007.
- Bollerslev, T. (1986): "Generalized Autoregressive Conditional Heteroskedasticity." Journal of Econometrics, 31(3), 307-327.

- Cryer, J. D., & Chan, K. S. (2008). Time series analysis: with applications in R (Vol. 2). New York: Springer.
- Katesari, H. S., & Vajargah, B. F. (2015). Testing adverse selection using frank copula approach in Iran insurance markets. Mathematics and Computer Science, 15(2), 154-158.
- Katesari, H. S., & Zarodi, S. (2016). Effects of coverage choice by predictive modeling on frequency of accidents. Caspian Journal of Applied Sciences Research, 5(3), 28-33.
- Safari-Katesari, H., Samadi, S. Y., & Zaroudi, S. (2020). Modelling count data via copulas. Statistics, 54(6), 1329-1355.
- Shumway, R. H., Stoffer, D. S., & Stoffer, D. S. (2000). Time series analysis and its applications (Vol. 3). New York: springer.